INCLUDING THE VIRTUAL LABORATORY CONCEPT IN AN ON-LINE COLLABORATIVE ENVIRONMENT

F. A. Candelas, F. Torres, P. Gil, S. Puente, J. Pomares

Automatics, Robotics and Computer Vision Group
Physics, Systems Engineering and Signal Theory Department
University of Alicante (Spain)

Abstract: With the aim of combining the main advantages of the virtual laboratories and the collaborative learning environments, a new on-line collaborative environment is proposed in this paper. This environment includes, in addition to other common tools, simulation applets in the user interface. To make possible the collaborative learning, the simulations work in a synchronized way for the users of a same group. The main features of the first prototype of the environment and its architecture are described. This prototype has been developed as a module for the popular CMS Moodle.

Keywords: Education, learning, group work, virtual, user interfaces, on-line, laboratory.

1. INTRODUCTION

Nowadays, the information and communication technologies, and specially the Internet, are extensively applied to education in many different ways. In the scope of the technical courses and careers, the use of virtual laboratories is very common (Candelas *et al.*, 2005). These laboratories cover from simple simulations to systems which allow the access to remote resources (Sánchez *et al.*, 2005; Candelas *et al*. 2003a). They make easy the experimentation and the practical learning of difficult concepts (Dormido 2003). Moreover, they provide many advantages, such as the flexibility of time-tables, the access to limited equipment, and the auto-evaluation systems (Candelas *et al*. 2003a).

Nevertheless, the majority of the virtual laboratories are designed to be used individually, and they do not permit the work in group and the collaboration among the students and the teacher. The collaboration and the communication are necessary in the learning process, in addition to individual work (Barros et al. 1998, Verdejo *et al.*, 2001). In previous academic years, the authors of this paper have evaluated the impact of the virtual laboratories which are used in their courses. These courses include topics such as Robotics, Computer Vision, Control Process and Automation. The main conclusion was that student considerers the virtual laboratory to be a valuable complement to the traditional teaching, but he also prefers to have a real laboratory at the University where they can work in coordination with their class-mates and have the support of a teacher (Candelas et al., 2003b).

In the last decade, many collaborative learning environments have been developed to include the collaboration and the communication in courses through Internet. These environments can be either on-line (synchronous) or off-line (asynchronous). The first ones permit the e-learning more similar to the traditional learning because they allow the students and the teacher to share the experiences in real-time during a lesson, while the second ones take advantage of very flexible time-tables.

The majority of the on-line collaborative environments developed use tools such chats, video streams or shared blackboards, which could be classified as classic, to communicate the members of a course. This is the case of the systems described in (Kreutz *et al.*, 2000; Isenhour *et al.*, 2000). But these tools limit the teaching on many technical courses, which could be explained more easily by means of demonstrations on a virtual laboratory. Some environments are more sophisticated and also allow the teacher to present slides on a whiteboard and even paint over it (Snow *et al.* 2005). However, very few on-line collaborative environments give support

to virtual laboratories. An example of this is the environment presented in (Sklar *et al*., 2000), which uses diverse multi-player educational games to teach topics such as geometry and grammar to children. But this environment is designed for primary school children.

Among the asynchronous environments, eMersion must be emphasized (Gillet *et al.*, 2005). This system provides simulation and teleoperation modules through the Web, which are used individually by students, although they can use a powerful system called eJournal to share their documents and reports.

In addition, other researches have tried to combine the use of conventional virtual laboratories with standard on-line communication tools. However, this usually is not a good solution because the user interface becomes very complex (Abler *et al.*, 2005).

Considering the previous aspects, our research group have designed a new on-line collaborative environment which incorporates shared modules of a virtual laboratory to improve the on-line classes. The environment has been called ClaseWeb (the Spanish word "clase" denotes the classroom, as well as the group of students which are in it). This environment will be incrementally incorporated in our technical courses through the Internet.

This paper is structured as follows. First, the main features and the architecture of the first prototype of ClaseWeb are described in Section 2. Next, in Section 3, the software architecture is explained with more detail. In Section 4, the user interface is described. Finally, Section 5 presents the conclusions of our work.

2. CLASEWEB

2.1. Design of ClaseWeb

We have considered that the main features that ClaseWeb has to offer are the following:

- The environment must enable the students and the teacher to develop an on-line classroom through the Internet, using WWW technology, because this technology is the more extended and accessible.
- It is necessary a constant connectivity among the students and the teacher during the class, with a continuous exchange of information, which allows them collaborate to resolve the problems presented.
- The user interface have to include shared modules specific to the subject, in addition to other components used in traditional environments, such as a chat, a shared blackboard or shared files.
- The environment must enable the teacher of a course to utilize the suitable modules in his course. The system should store a collection of existent modules, and the teacher could incorporate other new ones.
- Other common administration services like user management, schedulers, calendars, organization of the courses, etc. should also be kept in mind.

The modules to include in the interface can be simple simulations related to the topics studied in the classroom, or moreover, components which permit tele-operation of remote systems. To get the objective of an environment over WWW, we have chosen to develop these modules as Java applets (http://java.sun.com/).

To carry out a first prototype of ClaseWeb, we decided to incorporate it as a module inside an existent CMS (Content Management System). In this way, all the administration services are managed by the CMS, and the designers should work only with the specific aspects of the collaborative environment. More concretely, Moodle (http://moodle.org/) have been chosen because it is a very popular open-source CMS designed to teach courses through the Internet, and it is used by a great number of educational institutions around the word. It provides many important advantages, such as the possibility of including new modules and the good free support for its use and development.

2.2. Architecture

The fundamental elements of the ClaseWeb prototype are shown in the Figure 1. This on-line collaborative environment is provided by a server which executes, on the one hand, an active web server based on the APM architecture (Apache, PHP and MySQL), and, on the other hand, a server application programmed in Java and called SAS. The clients are web browsers with the necessary libraries to execute Java applets (JRE 1.5).

The web server also stores the Moodle's pages, including among them the module that implements the prototype of ClaseWeb. The SAS (Synchronized Applets Server) is a Java application which communicates and coordinates two components which are executed in each client as Java applets: a shared blackboard and a shared simulation.

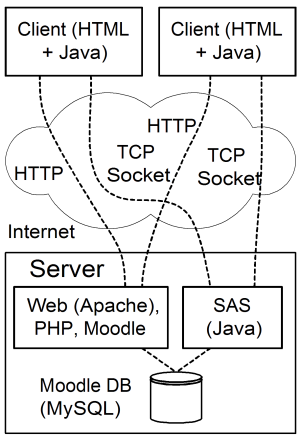

Fig. 1. Architecture of ClaseWeb.

At the beginning of the development, two options were discussed to coordinate the applets in the clients. The first option was the use of the Moodle's database and the web pages to share information among applets. The second option was to establish communication between each applet and a single server application (the SAS) which coordinates the set.

The principal troubles of the first option are the conversion and the storage of the exchanged Java objects as binary database objects, as well as, the necessity of including some functions in the applets to coordinate them. In contrast, the second option implies an easier and more independent implementation of the client applets, since the Java objects can be transmitted directly into java TCP/IP sockets. The more notable advantage of this solution is to use TCP/IP sockets, which involves reserving some TCP ports. In addition, the SAS must be executed in the same server where the Moodle pages are stored due to the security restrictions of the Java applets. To solve this restriction, the applets have to be signed.

The second option was chosen because the facility for adapting existing simulation applets to include them in ClaseWeb was considered a very important factor.

3. SYNCHRONIZED APPLETS

3.1. The Synchronized Applet Server (SAS)

The SAS is the application that manages the communication among the different Java applets which are executing in the client browsers of a same group of students and teachers. This communication is the base to synchronize the applets and make possible that they are in the same state and they show the same content. The SAS also manages the chalk assignment to a specific student in each classroom. This application is programmed in Java 1.5, and is executed as a background command-line program in the same computer where the web server is running.

The main objects which compose the SAS are shown in the Figure 2, and described next. The SAS executes a simple instance of an object Server, which includes an object Connection Handler and a vector of objects of the class Session. The Connection Handler attends new connection requests from the applets in the clients, and assigns them to the corresponding Session. Each Session attends to the set of applets which belong to the same group of students and teacher. Moreover, for the same group, two sessions are created, one to coordinate the applets of the shared blackboards, and other one to coordinate the simulation applets. A new Session is created if the connection request comes from the first applet which is executed in a group.

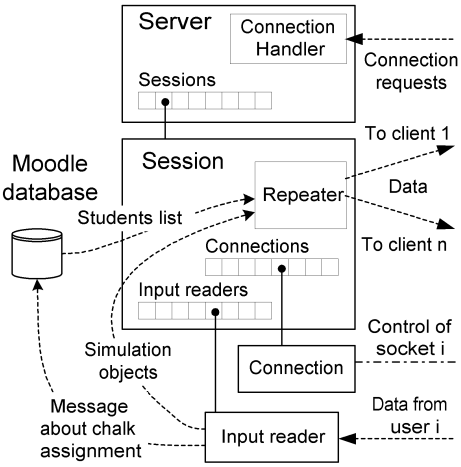

Fig. 2. Basic structure of the SAS.

Inside the Moodle environment, several instances of the ClaseWeb module can be included in different courses. For this reason, many instances of an object Session could be executed at the same time.

Each Data exchanged between a Session and an applet is an object composed of a label, a user identifier, a session identifier, a serializable object of Java and its size. The label is used to identify the class of the object, and the user identifier represents the student or teacher which uses the client. With the object Data, simple objects of Java as a String, an Integer or a Double can be transmitted. Also arrays of these classes can be used. These classes of objects are sufficient for exchange the state of the applets, and obtain that they be synchronized, although new serializable objects might be programmed.

For example, an object Data can be an array with two integers, that represents a coordinate "x,y" of a simulation. The corresponding label could be "XY-Coordinate". In addition, other control messages between the SAS and the applets are represented using these objects, as for example the chalk assignment to an applet in a student's client, or the list of students sent to the applet of a teacher.

The Data objects received by a Session from an applet are retransmitted to the rest of applets by means of the object Repeater. The Session also explores the label of the Data object to find the messages of chalk assignment among the rest. This kind of messages implies that the SAS writes a text about the chalk changing in the database of Moodle in order to generate a message in the chat associated to the session. This message will be presented in the clients the next time that they update automatically the frame associated to the chat.

When a new client is connected or disconnected, the Session access to the Moolde's database to get the current students lists of the group, and send it to the user interface of the teacher's applet, where a control with the list is updated. This control allows the teacher to choose a student to send him the chalk.

3.2. The client structure

The two applets executed in the user's client (the simulation and the blackboard applets) need to communicate with the SAS, and they require some communication functions, as it has been described previously. It is desirable that existent applets can be easily included in the environment, with the minimum changes. Because it, the functions have been programmed in a class to make it transparent to the original applet. This class is called Applet1, and must be added to the original applet following the structure shown in the Figure 3.

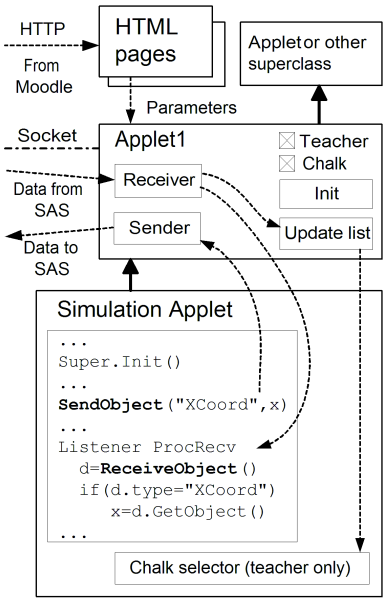

Fig. 3. Structure of a simulation applet.

The same class structure is applied to both, the simulation applet and the blackboard applet. But the second one has been included yet in the Moodle's module directory, and it does not require any change. Therefore, this section describes how to include a new simulation applet in the environment.

The object Applet1 receives from the web page some necessary parameters such as the user identifier, the kind of user (student or teacher) and the communication parameters of the server and the SAS. Then, it creates a TCP socket between it and the SAS, and it starts the simulation applet and waits for data from the SAS and actions from the user interface.

Applet1 should extend the base class of the original simulation applet (for example the class Applet of Java), and the applet have to be modified to extend Applet1. After this, the communication and chalk management features will be available in the applet. Next, some modifications are necessary to adapt the original simulation applet, as follows.

First, the simulation applet must be able to send data to the CAS when an important simulation parameter is changed. This usually success when an event of the user interface is triggered. To achieve this, the function SendObject has to be used inside the necessary events. For example, when a slide button is clicked to set a value, the SendObject function will be included in the corresponding event to send the value as a Data object (see section 3.1) with the label "XCoord".

Second, it is necessary to add to the simulation applet an event manager (listener) for the received Data from the SAS. This listener must update the state of the applet, or activate some events, according to the received data. For example, if a Data received has the label "XCoord", the corresponding value of the applet must be update, in the same way that the user interface generates the event of the slide button.

Finally, two management functions must be included. On the one hand, if the applet belongs to a student, it must allow the user to activate user interface events only when the applet has the Chalk indicator active. This indicator becomes active when a chalk is received form the CAS. On the other hand, if the applet owns to a teacher and it receives a Data object with the students list, it must update a control with this list in the user interface. This control enables the teacher to send the chalk to the desired student.

Following the previous steps, we have included in the prototype of ClaseWeb not only independent simulation applets, but also applets created with the software Easy Java Simulations (EJS, Esquembre, 2005; Sánchez *et al.*, 2005). In this case, the Applet 1 class must be placed between the class "org.opensourcephysics.ejs.LauncherApplet" and the class that implements the simulation applet. Also the necessary *.jar libraries of EJS must be allocated in the web server.

3.3. Chalk management

In a session of ClaseWeb, all the blackboard applets share a virtual chalk which indicates what student can modify the applet's state, for example, painting a drawing. All the simulation applets also share other chalk in the same way, which enables the corresponding user to modify the simulation controls or activate the buttons.

When a new session of ClaseWeb is started, the chalks are assigned to the first teacher that connects to the session. The teacher has the privilege of assign the chalk to the desired student in any time. Thus, it is necessary that the teacher initiates a session in the module to begin the class and pass the chalk to the students.

To assign the chalk, the applet of the teacher's client sends a Data object to the SAS that specifies a student (See Figure 4). The SAS interprets the Data object and retransmits it to the corresponding student applet. In addition, the SAS insert a new message in the database associated to the chat of the session which describes the chalk assignment.

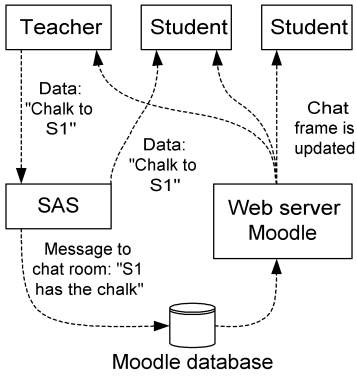

Fig. 4. Chalk assignment steps.

4. USER INTERFACE

After the web administrator has installed the ClaseWeb module in Moodle, the first step that a teacher has to do to use this module is to include it in a theme or topic of an existing course in Moodle. In this time, the teacher can select the simulation applet that he desires to use, and choose between to use or not the shared blackboard. The compiled java applet (*.class and *.jar files) must have been previously uploaded into a folder of the course. The ClassWeb module will copy the applet files into a public folder of the web, in order to make it accessible. After this step, the module is available in the corresponding topic of the course, as the example that the Figure 5 shows (see red arrow).

Not only different instances of the ClaseWeb module can be included in different courses, but also several instances of the ClaseWeb module can be included in different topics of the same course.

When a user (teacher or student) access to the module, the structure of HTML frames and pages which composes the user interface is loaded in the client browser. The Figure 6 shows an example of this structure. The thin upper frame shows the context of the pages inside the Moodle web, with the corresponding links. Under the context frame is the largest frame, which contains the applets with the shared blackboard and the simulation. The shared blackboard has controls to draw in different colours, erase a part o the whole drawing, and paint texts.

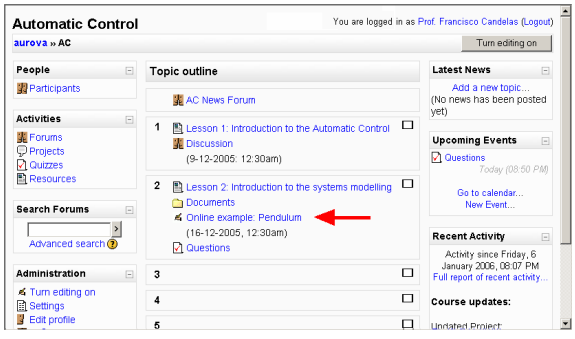

Fig. 5. The ClaseWeb module inside a Moodle's course.

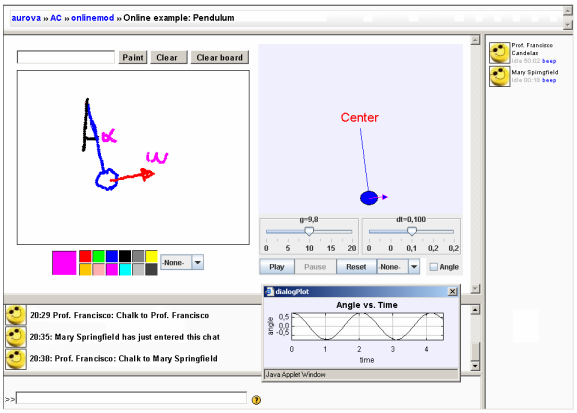

Fig. 6. User interface of the ClaseWeb prototype.

In the example shown in Figure 6, a draft of a pendulum was drawn in the blackboard, and the simulation is an applet that allows the users to modify and test the parameters of a pendulum. This applet has been created using the software Easy Java Simulations (Esquembre, 2005; Sánchez 2005), and modified to communicate the simulation parameters to the other clients, as it was described in section 3.2. In the example, the simulation applet has opened a window where the angle of the pendulum is plotted.

When the user is a teacher (or administrator) the two applets have in their interfaces a control (a combo-box) that the teacher can use either to pass the chalk to a connected student, or to withdraw the chalk (selecting "none"). This is the case shown in Figure 6. Students can not see these controls.

Under the frame with the applets, there is other frame which lists the messages sent to the chat associated to the instance of the collaborative module. Among these messages are the alerts about the change of the chalk. Under this frame there is another in which a user can write a new message to the chat.

Finally, the frame on the right side of the browser window lists the users which are using the module.

5. CONCLUSIONS AND FUTURE WORK

The main achievement of the first prototype of ClaseWeb is to allow the teacher and the students of an on-line course through the Internet to use shared simulations in order to experiment practical concepts in a coordinated way. The use of a simulation is a more efficient method than the familiar chat and blackboards modules to teach many engineering topics. In addition, the prototype allows the teachers to include in their courses many existent applets with some changes in it. For example, the possibility of including applets created previously with the software Easy Java Simulations has been shown.

Many proposed on-line collaborative systems execute independent applications (such as a video-conference software, applications for systems

simulation, and a Web browser) in the same computer's desktop, and this usually makes difficult to use the interface. In contrast, a system like ClaseWeb provides an integrated interface which includes a simulation apart from classic tools, like a blackboard and a chat. The fact of including all the necessary tools in the same interface is a great advantage because it facilitates the access to all the user-interface capabilities of these tools.

To create the on-line virtual learning environment, it has not been necessary develop the whole system thanks the use of the development capabilities of Moodle. A CMS like this offers all the necessary tools for the management of courses through the Internet. Our work has been centred in the design of the on-line collaborative environment.

We also should point out that the server of the current prototype is executing on a Linux platform, although other OS could be used if it has installed the APM software and the Java 1.5 runtime. Apart from the OS and Java, the used software (web server, database, and Moodle) is popular free software. The system will be available in our web server (www.aurova.ua.es).

The next work that we will have to do is evaluate the effect of ClaseWeb in the e-learning, using it to teach practical exercises through Internet. We are also studying how to improve the structure of the client applet to make easier the incorporation of the necessary communication functions in an existent applet. The final objective is to include, not only simulation applets, but also applets which allow students to tele-operate remote laboratories.

REFERENCES

Abler, Randal T., Wells, I. Gail (2005) Distributed Engineering Education: Evolution of the Telecollaboration Stations for Individualized Distance Learning. *IEEE Transactions on Education* (IEEE Education Society), **48,3**, 490-496.

Barros, B., Verdejo, M.F., and Rodriguez-Artacho, M. (1998). Towards a model of collaborative support for distance learners to perform joint tasks. *The Virtual Campus. Trends for Higher Education and Training* (Chapman & Hall Ed.), 155-168.

Candelas, F. A., Puente, S. T., Torres, F., Ortiz, F. G., Gil P. and Pomares, J. (2003a). A Virtual Laboratory for Teaching Robotics. *International Journal of Engineering Education*, Special issue "Remote Access / Distance Learning Laboratories" (Tempus Pub.), **19,3**, 363-370.

Candelas, F. A., Torres, F., Ortiz, F. G., Gil, P., Pomares, J. and Puente, S.T. (2003b). Teaching and Learning Robotics with Internet Teleoperation. *Proceedings 2nd Int. Conference on Multimedia and Information & Communication Technologies in Education (m-ICTE 2003)*, Badajoz (Spain), **3**, 1827-1831.

Candelas, F. A. and Sánchez, J. (2005). Recursos Didácticos Basados en Internet para el Apoyo a la Enseñanza de Materias del Área de Ingeniería de Sistemas y Automática. *Revista Iberoamericana de Automática e Informática Industrial (RIAI)* (CEA-IFAC)), **2,2**, 93-101.

Dormido, S. (2003). The role of Interactivity in Control Learning. *Proceedings of the 6th IFAC Symposium on Advances in Control Education*, Oulu, Finland, 2003, pp. 11-22.

Esquembre, F. (2005) Easy Java Simulations (http://www.um.es/fem/Ejs/)

Gillet D., Nguyen Ngoc, A.V. and Rekik, Y. (2005). Collaborative Web-Based Experimentation in Flexible Engineering Education. *IEEE Transactions on Education* (IEEE Education Society), **48,4**, 696-704.

Isenhour, P.L., Carroll, J.M., Neale, D.C., Rosson, M.B. and Dunlap, D. R. (2000). The Virtual School: An integrated collaborative environment for the classroom. *Journal of Educational Technology & Society*, Special issue "On-line Collaborative Learning Environments" (International Forum of Educational Technology & Society), **3,3**, 74-86.

Kreutz, R., Kiesow, S. and Spitzer, K. (2000) NetChat: Communication and Collaboration via WWW. *Journal of Educational Technology & Society*, Special issue "On-line Collaborative Learning Environments" (International Forum of Educational Technology & Society), **3,3**, 87-93.

Sánchez, J., Esquembre, F., Martín, C., Dormido, S., Dormido, R., Dormido, S., Pastor, R. (2005). Easy Java Simulations: An Open-Source Tool to Develop Interactive Virtual Laboratories Using Matlab/SImulink. *Int. Journal of Engineering Education* (Tempus Pub.), **21,5**, 798-813.

Sklar, E. and Pollack, J. (2000). A Framework for Enabling an Internet Learning Community. *Journal of Educational Technology & Society*, Special issue "On-line Collaborative Learning Environments" (Int. Forum of Educational Technology & Society), **3,3**, 393-408.

Snow, C., Pullen, M. and McAndrews, P. (2005). Network EducationWare: An Open-Source-Web-based System for Synchronous Distante Education. *IEEE Transactions on Education* (IEEE Education Society) **48,4**, 705-712.

Verdejo, M.F., Barros B., and Rodríguez-Artacho, M. (2001). A proposal to support the design of experimental learning activities. *Proc. European Conference on Collaborative Learning 2001*, Maastricht (The Netherlands), 633-640.